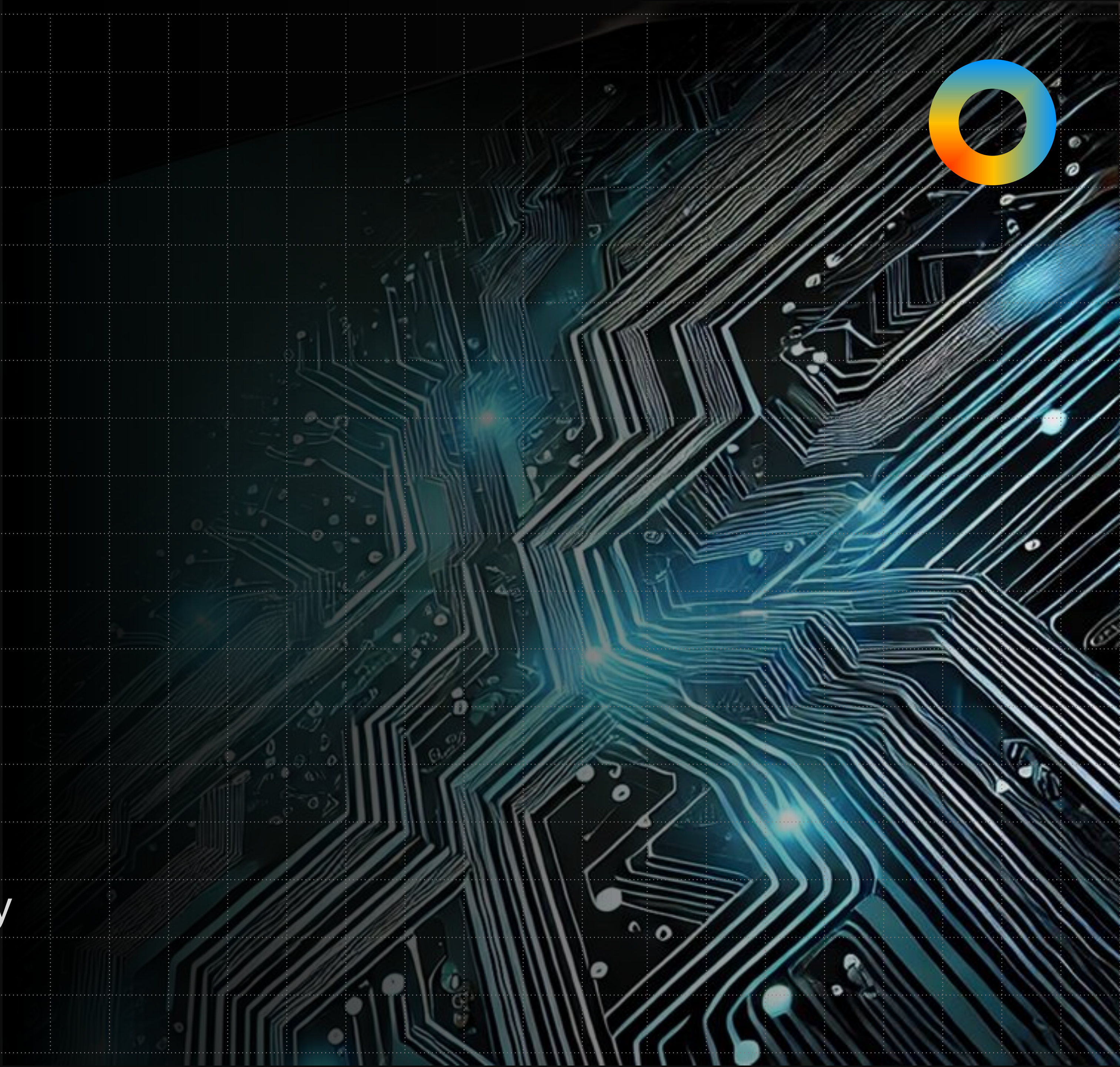
Python Programming

Functions

Dr. Chun-Hsiang Chan
Department of Geography
National Taiwan Normal University



Outlines

- Functions
- Recursive Functions

Functions

```
def function_name(input_arg):
```

- Sometimes, you have to do something many times; however, there is no built-in package or function that helps you.
- As a result, you need to design the customized function by yourself.

```
# typical function
def cm2m(cm):
    m = cm/100
    return m

# use defined function
cm2m(120)
```

```
# simple function
def cm2m_(cm):
    return cm/100

# use simple function
cm2m_(120)
```

Functions

- Here, we introduce ”local variable” and “global variable”.
- All variables in the function indentation are local variables which indicates that they cannot be used outside the block.
- Meanwhile, the all variables used outside the function cannot be used in the function.

global

```
# observe the variables
cm = 1000
m = 900
a = 25
print(cm, m, a)
```

local

```
def cm2m(cm):
    global a, x
    a, x = 49, 13
    m = cm/100
    print(cm, m, a, x)
    return m
```

global

```
cm2m(120)
print(cm, m, a, x)
```

Functions

- Why do we need a local variable?

- Why do we need a global variable?

- Please give the reason with examples.

Functions

- You may design **multiple inputs** for a function.

```
# multiple inputs
def affiliation(name, dept, institution):
    nameInfo = 'Dr. '+name+' at '+dept+', '+institution
    print(nameInfo)
    return nameInfo

# try it
text = affiliation('CCH', 'Dept. of Geography', 'NTNU')
```

Functions

- You may also design **multiple outputs** for a function.

```
# multiple outputs
def degCTransform(degreeC):
    degreeK = degreeC + 273.15
    degreeF = degreeC * (9/5) + 32
    return degreeC, degreeK, degreeF

# try it
C, K, F = degCTransform(25)
```

Functions

- In some scenarios, you may be unsure of the exact number of arguments a function will need to handle.

```
# unknown arguments
def hello(**kwds):
    print("Hello " + kwds["fname"] + kwds["lname"])
    return None

# try it
hello(lname = "Chan", fname = "C.H.", dept = "Geo")
```

Functions

- Sometimes, when defining a function, we specify a default argument that is used if no explicit input is provided.

```
# default argument
def sayHello(name = "Everyone"):
    print("Hello " + name)
    return "Hello " + name

# try it
sayText1 = sayHello("Tom")
sayText2 = sayHello()
```

Recursive Functions

- Recursive function is a powerful approach to get some results with special rules or regularities.

```
# recursive function
def my_sum(a):
    if a == 1 or a == 0:
        return a
    else:
        return a + my_sum(a-1)

# use recursive function
my_sum(10)
```

```
# if a = 4, then …
4 + my_sum(4–1) # 4 – 1 = 3
4 + (3 + my_sum(3–1))
4 + (3 + (2 + my_sum(2–1)))
# which is 1
# Because … my_sum(2–1) = 1
# So …
4 + (3 + (2 + 1)
4 + (3 + 3)
4 + 6
10
```

Lab Practice #1 (recursive function)

- Design a function that can calculate the **factorial** answer.
- Example: **my_factorial(5) = 120**

```
# recursive function
def my_factorial(a):
    ...
    ...
    ...
    ...
```

Lab Practice #2 (recursive function)

- Design a function that generates a **fibonacci** number.
- **my_fibon(10)** # 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, …

```
# recursive function
def my_fibon(a):
    …
    …
    …
    …
```

The End

Thank you for your attention!

Email: chchan@ntnu.edu.tw

Website: https://toodou.github.io/

STDS Lab
Est. 2022